5

ÖSTERREICHISCHES PATENTAMT

A-1014 WIEN, KOHLMARKT 8 – 10

Kanzleigebühr € 15,00
Schriftengebühr € 65,00

Aktenzeichen **A 338/2002**

Das Österreichische Patentamt bestätigt, dass

Mag. René - Michael Cordes
in A-2323 Mannswörth, Raiffeisengasse 3
(Niederösterreich),

am **5. März 2002** eine Patentanmeldung betreffend

"Codegenerator und Vorrichtung zur Synchronen oder Asynchronen sowie permanenten Identifikation oder Ver- und Endschlüsselung von Daten beliebiger Länge",

überreicht hat und dass die beigeheftete Beschreibung samt Zeichnungen mit der ursprünglichen, zugleich mit dieser Patentanmeldung überreichten Beschreibung samt Zeichnungen übereinstimmt.

Österreichisches Patentamt
Wien, am 14. März 2003

Der Präsident:

i. A.

K. BRUNŽAK

Mag. René - Michael Cordes

A 338/2002

Urtext

BESCHREIBUNG

Alle Verschlüsselungsmethoden benützen einen Code, auch wenn die zu verschlüsselnde Information selbst als Code Verwendung findet.

Je besser der Code versteckt ist desto effektiver ist die Verschlüsselung.

Je länger der Code ist desto schwerer ist er zu entschlüsseln.

Ein unendlicher Code bräuchte gar nicht versteckt zu werden, da er ja nie ganz bekannt ist.

Funktionell ist jeder Code als unendlich anzusehen, der sich nicht vor dem Ende der zu verschlüsselnden Information wiederholt

Ein funktionell unendlicher Code hat den Vorteil, dass der Ver- und Entschlüsselungsvorgang selbst denkbar einfach durch eine einzige EXOR – oder EXNOR – Verknüpfung realisiert werden kann.

Ein funktionell unendlicher Code hat den Nachteil, dass er nicht übertragen werden kann. Er muss generiert werden

Damit so ein Code nicht nachgeneriert werden kann muss der Generator so geartet sein, dass er so viele verschiedene Codes erstellen kann, dass von einem Teil des einzelnen Codes nicht auf dessen Fortsetzung geschlossen werden kann.

Nachstehende Darstellung kann entnommen werden, dass all diese Anforderungen von der Erfindung erfüllt werden.

Wenn man zwei FLIP FLOP {FF1, FFn} (74HC174) und ein EXOR - Gatter {EXOR1} (74HC386,) und einen Inverter {INV} (4009) in der Weise verschaltet, dass der Eingang (74HC386, 1) des EXOR - Gatters {EXOR1} und der Eingang (74HC386, 2) des EXOR - Gatters {EXOR1} jeweils mit einem der beiden Ausgänge der beiden FLIP FLOP [{FF1} (74HC174, 2), {FFn} (74HC174, 5)] und der Ausgang des EXOR - Gatters {EXOR1} (74HC386, 3) mit dem Eingang des zweiten FLIP FLOP {FFn} (74HC174, 3) und der Ausgang des zweiten FLIP FLOP {FFn} (74HC174, 5) mit dem Eingang des Inverters {INV} (4009, 9) und der Ausgang des Inverters {INV} (4009, 2) wiederum mit dem Eingang des ersten FLIP FLOP {FF1} (74HC174, 3) - sohin rekursiv - verbindet, erhält man eine Schaltung [Schaltung: Fig. 1], die eine Abfolge von HIGH und LOW Level (im Weiteren Code genannt) erzeugt.

Es vergehen 3 Takte ehe sich der Code wiederholt.

Wenn zwischen diesen beiden FLIP FLOP {FF1, FFn}eine Anzahl von weiteren FLIP FLOP {FF2, FF3, ...} geschaltet werden [Wobei eine Anzahl ununterbrochen hintereinander geschalteter FLIP FLOP auch in Form von Schieberegister {SRG1, SRG2, ...} (74HC164) verwirklicht werden kann], so verdoppelt sich die Länge des Codes pro weiteren FLIP FLOP, so dass sich die Länge des Codes wie folgt berechnet [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

$$l = 2^n - 1$$

(l = Länge der Abfolge; n = Anzahl der codegenerierenden hintereinander geschalteten FLIP FLOP)

Wenn diese Einheit mit einem bestimmten Takt betrieben wird gilt für die Dauer des Codes:

$$T_c = \frac{2^n - 1}{f_c}$$

(T_c = Dauer bis sich der Code wiederholt; f_c = Codegenerierungstaktfrequenz)

Wenn man nun an mehren Stellen dieser FLIP FLOP Kette ein EXOR – Gatter einfügt und solcherart mehrere rekursive Stellen einbaut so verändert man jeweils den dadurch erzeugten Code [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

Wenn man an einen der Beiden Eingänge des jeweiligen EXOR - Gatters {EXOR1,2,..,n} den Ausgang eines UND - Gatters {UND1,2,..,n} dessen einer Eingang am Ausgang des FLIP FLOP {FFx} hängt, anschließt, dann kann man diese rekursiven Stellen über den zweiten Eingang des UND- Gatters {UND1,2,..,n} an- und abschalten und wenn man daran jeweils ein weiteres FLIP FLOP {FFp1,p2,..,np} anschließt, das Abschalten der rekursiven Stellen programmierbar machen [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

Die Anzahl der programmierbaren unterschiedlichen Codes berechnet sich wie folgt:

$$N_c = 2^{pn} - 1$$

(N_c = Anzahl der möglichen unterschiedlichen Code; pn = Anzahl der programmierbaren rekursiven Stellen)
Daraus ergibt sich eine Wahrscheinlichkeit bei kontinuierlicher Beobachtung des Codes die dem Code zugrundeliegende Programmierung zu erkennen und sohin den weitere Verlauf des Codes vorauszusagen von:

$$w = \frac{N_b * f_c}{(2^n - 1) * (2^{pn} - 1)}$$

(N_b = Anzahl der beobachteten Bit der Codesequenz; f_c = Codegenerierungstaktfrequenz; n = Anzahl der codegenerierenden hintereinander geschalteten FLIP FLOP; pn = Anzahl der programmierbaren rekursiven Stellen)

Zur Vermeidung von codelängenreduzierenden Resonanzeffekten ist es zweckmäßig die verschiedenen Rückbezugsstellen in unterschiedlichen möglichst dissonanten Entfernungen voneinander einzubauen so etwa jeweils an den FLIP FLOP deren Position einer Primzahl entspricht.

Indem man nur jeden 2. Zustand eines Muttercodes einen von zwei Tochtercodes zuweist kann man aus einem zwei Codes schaffen, welche zwar verwandt aber nicht ähnlich sind [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

Die Codierung des Eingangssignals erfolgt durch ein EXNOR Gatter (74HC266) in dessen einen Eingang das zu verschlüsselnde Signal und in dessen zweiten Eingang der Code eingebracht wird, so dass an dessen Ausgang das mit dem Code verschlüsselte Ausgangssignal erscheint .

Die Decodierung des Eingangssignals erfolgt durch ein EXNOR Gatter (74HC266) in dessen einen Eingang das zu entschlüsselnde Signal und in dessen zweiten Eingang der Code eingebracht wird, so dass an dessen Ausgang das mit dem Code entschlüsselte Ausgangssignal erscheint.

All diese Elemente können zu einer Funktionseinheit zusammengefasst werden, welche im weiteren Schlüssel genannt wird [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

Da die Schaltung bei der Codegenerierung selbst keine CPU Zeit in Anspruch nimmt, ist sie unabhängig von jedweder Hand Shake Zeit und daher einzig und allein durch die spezifischen Schaltzeiten, der elektronischen Bauelemente, aus denen sie aufgebaut ist, in ihrer Codeproduktionsgeschwindigkeit begrenzt. (Mit handelsüblichen TTL Bauelementen sind ohne weiters Codeproduktionsgeschwindigkeiten im Megaherzbereich realisierbar)

Innerer Ablauf der Schaltung		
AKTION	REAKTION	FUNKTION
1.) Der Schlüssel A wird mit seinem Ausgang für Computer und Schlüssel in den Eingang für Schlüssel des Schlüssels B hineingesteckt	Sowohl in Schlüssel A als auch in Schlüssel B werden die jeweiligen Kontakteingänge auf LOW gesetzt	Die Produktion von Code wird in Schlüssel A und B eingestellt Schlüssel A wird von seinem eigenen Takt betrieben Der Takt wird von Schlüssel A auf Schlüssel B übertragen
2.) Die Riegeltaste (Riegel AUF, Riegel ZU) des Schlüssel B wird erstmals betätigt	Das Riegel AUF – Signal wird von Schlüssel A nach Schlüssel B weitergeleitet In beiden Schlüsseln wird ein CLR – Signal abgeleitet	Sämtliche Schieberegister in den beiden Schlüsseln werden gelöscht
3.)	Der Takt wird in den Programmierteil sowohl von Schlüssel A als auch von Schlüssel B eingespeist	Die Programmierung läuft verzahnt synchron in beiden Schlüsseln, jedoch ohne direkte Übertragung der Programmierung von Schlüssel A nach Schlüssel B, ab.
4.) Die Riegeltaste des Schlüssel A wird abermals betätigt	Der Takt wird in den Programmierteil sowohl von Schlüssel A als auch von Schlüssel B nicht mehr eingespeist	Synchrones Stopp der Programmierung
5.) Die beiden Schlüssel werden getrennt	Sowohl in Schlüssel A als auch in Schlüssel B werden die jeweiligen Kontakteingänge auf HIGH gesetzt	Die Produktion von Code kann nun wird sowohl in Schlüssel A und B synchron gestartet werden (optional).

Mit Hilfe zweier solcher Schlüssel lässt sich nachstehende Anwendungen verwirklichen:

Übersicht der möglichen Abläufe			
MODUS	**Identifikationsmodus**	**Synchronisationsmodus**	**Synchronmodus**
CODE	X	X	X
ADRESSE		X	X
ZEIT			X
Identifizieren	X		
Synchronisieren	X	X	
Decodieren	X	X	X

Identifikationsmodus					
(A)	(B)	PHASE	HANDLUNGEN	SCHLÜSSEL (A)	SCHLÜSSEL (B)
-x	-x	Programmierung	Schlüssel (A) wird mit seinem Computer/Schlüssel Ausgang in den Schlüsseleingang von Schlüssel (B) hineingesteckt	Schlüssel (A) wird programmiert und wird ihm der Code C1 als Sendecode und der Code C2 als Empfangscode zugewiesen	Schlüssel (B) wird programmiert und wird ihm der Code C2 als Sendecode und der Code C1 als Empfangscode zugewiesen
0	0	Trennung	Schlüssel (A) wird aus Schlüssel (B) herausgezogen	Schlüssel (A) schweigt	Schlüssel (B) schweigt
0	0	Schlüsselaktivierung (A)		Schlüssel (A) wird in einen Computer gesteckt	
0	0			Eine E-Mail wird auf dem Computer von Schlüssel (A) geschrieben	
0 + 1000 + E – Mail	0	Codepaketabruf (A, 1000 + E- Mail)		Die E-Mail wird (life) mit Hilfe von C1 codiert, wobei eine Sequenz von 1000 Bit leeren Codes samt der Codeentstehungszeit uncodiert dem E-Mail vorangestellt wird	
0 + 1000 + E – Mail	0	Datenübertragung		Die E-Mail wird an die Homepage des Computers von Schlüssel (A) gesandt und dort ausgestellt	
0 + 1000 + E – Mail	0	Schlüsselaktivierung (B)			Schlüssel (B) wird in einen Computer gesteckt
0 + 1000 + E – Mail	0 + 1000	Codepaketabruf (B, 1000)			Eine Sequenz von 1000 Bit des Code C1 wird aus Schlüssel (B) ausgelesen und an eine Suchmaschine transferiert
0 + 1000 + E – Mail	0 + 1000				Wenn die Homepage von Schlüssel (A) entdeckt ist wird die E-Mail abgerufen
0 + 1000 + E – Mail	0 + 1000 + E – Mail	Codepaketabruf (B, E- Mail)			Schlüssel (B) decodiert (asynchron) mit Hilfe von C1 das Signal von Schlüssel (A)

0 + 1000 + E – Mail	A	Codefreifluss (B)			Schlüssel (B) generiert nunmehr laufend Code
0 + 1000 + E – Mail	A				Schlüssel (B) weiß jetzt wo Schlüssel (A) zu finden ist und kann mit ihm Verbindung aufnehmen um sich mit ihm zu synchronisieren.
Synchronisationsmodus					
B	0				Die E-Mail wird vom Server des Schlüssel (B) abgerufen
C	0 + 1000	Codepaketabruf (B, 1000)			Schlüssel (B) vergleicht die 1000 Bit Sequenz von Schlüssel (A) mit seinem C1 und verschiebt diese so lange bis Synchronität besteht (erkennbar an den leeren 1000 Bit) decodiert (synchron) mit Hilfe von C1 das Signal von Schlüssel (A) Ergebnis: leer
D	0 + 1000 + E – Mail	Codepaketabruf (B, E- Mail)			Schlüssel (B) decodiert (synchronisiert) mit Hilfe von C1 den Inhalt der Botschaft von (A)
E	E	Codefreifluss (A, B)			Schlüssel (B) weiß jetzt wo sich Schlüssel (A) auf der Zeitachse befindet, so dass er nunmehr synchron mit ihm Code aussenden kann und solcherart in den Synchronmodus wechseln kann.
Synchronmodus					
E + 1000	E + 1000			Schlüssel (A) decodiert (life) mit Hilfe von C2 das Signal von Schlüssel (B) Ergebnis: leer	Schlüssel (B) decodiert (life) mit Hilfe von C1 das Signal von Schlüssel (A) Ergebnis: leer
F + Botschafts dauer	F + Botschafts dauer	Datenübertragung	Ein Inhalt wird in Schlüssel (A) eingespeist	Schlüssel (A) codiert (life) mit Hilfe von C1 den Inhalt der Botschaft	Schlüssel (B) decodiert (life) mit Hilfe von C1 den Inhalt der Botschaft von (A)
G + Botschafts dauer	G + Botschafts dauer		Ein Inhalt wird in Schlüssel (B) eingespeist	Schlüssel (A) decodiert (life) mit Hilfe von C2 den Inhalt der Botschaft von Schlüssel (B)	Schlüssel (B) codiert (life) mit Hilfe von C2 den Inhalt der Botschaft

Mag. René - Michael Cordes

PATENTANSPRÜCHE:

Codegenerator, dadurch gekennzeichnet, dass

1. zwei oder mehr FLIP FLOP (74HC174) und ein oder mehr EXOR - Gatter (74HC386) und ein oder mehr UND - Gatter (74HC08) und ein Inverter (4009) in der Weise verschaltet sind, dass ein Eingang des EXOR - Gatter {EXOR1,2, ... n} mit dem Ausgang des FLIP FLOP {FF1,2, ... n} und der zweite Eingang des EXOR - Gatter {EXOR1,2, ... n} mit dem Ausgang des UND - Gatter {UND1,2, ... n} dessen einer Eingang mit einem Ausgang eines FLIP FLOP {FFp1,2, ... pn} und dessen zweiter Eingang mit dem Ausgang eines in der Kette befindlichen FLIP FLOP {FFx}und der Ausgang der EXOR - Gatter {EXOR1,2, ... n} mit dem Eingang des FLIP FLOP {FF2,3, ... n} und der Ausgang des in Flussrichtung letzten FLIP FLOP {FFn} in der Kette mit dem Eingang des Inverters {INV} und der Ausgang des Inverters wiederum mit dem Eingang des in Flussrichtung ersten FLIP FLOP {FF1} in der Kette - sohin rekursiv - verbunden ist [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

2. er keine CPU Zeit in Anspruch nimmt und sohin unabhängig von jedweder Hand Shake Zeit, einzig und allein durch die spezifischen Schaltzeiten, der elektronischen Bauelemente, aus denen er aufgebaut ist, in seiner Codeproduktionsgeschwindigkeit begrenzt ist.

3. die Länge des von ihm ohne Wiederholungsrunden produzierten Codesequenz sich wie folgt berechnet:

$$T_o = \frac{2^n - 1}{f}$$

(T_c = Dauer bis sich der Code wiederholt, n = Anzahl der in Reihe geschalteten FLIP FLOP, f = Codegenerierungstaktfrequenz (Mit weniger als 50 FLIP FLOP bei einer Codegenerierungstaktfrequenz von 384.000 Bit/s läuft der Code länger als ein Jahr ohne dass sich die Sequenz wiederholt, so dass ein zu verschlüsselndes Signal simultan über einen ebenso langen Zeitraum verschlüsselt, über eine Standleitung übersendet und entschlüsselt werden kann, so dass verschlüsselte Lifeübertragungen über einen ebenso langen Zeitraum möglich sind.)

4. der generierte Code über den Zustand der FLIP FLOP {FFp1, p2, ... pn} in seinem Inhalt auf $2^{pn} - 1$ verschiedene Arten programmierbar ist, so dass auch bei kontinuierlicher Beobachtung des Codes nur mit der Wahrscheinlichkeit von

$$w = \frac{N_b * f_c}{(2^n - 1) * (2^{pn} - 1)}$$

(w= Wahrscheinlichkeit der Erschließung der Programmierung; N_b = Anzahl der beobachteten Bit der Codesequenz; f_c = Codegenerierungstaktfrequenz; n = Anzahl der codegenerierenden hintereinander geschalteten FLIP FLOP; pn = Anzahl der programmierbaren rekursiven Stellen)

die dem Code zugrundeliegende Programmierung erschließbar und sohin der weitere Verlauf des Codes voraussagbar ist.

Mag. René - Michael Cordes

5. die Programm FLIP FLOP {FFp1, p2, p3, ... pn} ebenfalls in unter Punkt 1. beschriebener Weise rekursiv miteinander verschaltet sind [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig.3], so dass sie innerhalb des Zeitintervalls

$$T_{pn} = \frac{2^{pn} - 1}{f_p}$$

(T pn = Durchlaufzeit aller möglichen Programmierzustände; pn = Anzahl der Programm FLIP FLOP; f_p = Programmiertaktfrequenz) sämtliche mögliche Zustandskombinationen durchlaufen, so dass die Programmierung sich aus einer bestimmte Zeitspanne ergibt, in der die Programm FLIP FLOP mit einem Programmtakt versorgt werden, so dass durch gleichzeitiges Ein- und Ausschalten des Programmtaktes an zwei identen Schlüsseln diese so durchgeführt werden kann, dass wohl mehrere idente Schlüssel entstehen der Innhalt der Programmierung aber nicht einmal den Programmierern bekannt ist

6. durch Nutzung von lediglich einem Teil der Codesequenz (zum Beispiel jeweils nur jedes 2. Bit) man abgeleitete Codesequenzen erzeugen kann, die zwar zueinander verwandt, aber nicht einmal ähnlich sein müssen

7. durch Einsatz von wenigstens zwei ihrer Art in der in Schaltung Fig. 2 dargestellten Vorrichtungen (Schlüssel) ein zweiter Inhaber desselben Schlüssels identifiziert werden und in weiterer Folge durch Synchronisation der Schlüssel mit diesem eine ständige, verschlüsselte Datenverbindung aufgebaut werden kann, über die permanent und life Daten ausgetauscht werden können.

Mag. René - Michael Cordes

ZUSAMMENFASSUNG

Codegenerator und Vorrichtung zur synchronen oder asynchronen sowie permanenten Identifikation oder Ver- und Entschlüsselung von Daten beliebiger Länge, welcher auf die Erzeugung von 2^{pn} unterschiedlichen 2^{n} lange Codesequenzen programmierbar ist, welche so zahlreich sein können, dass die Wahrscheinlichkeit den richtigen zu finden unter $1:1*10^{10}$ liegen kann und welche so lange sein können, dass bei einer Codeproduktionsgeschwindigkeit von 384.000 Bit/s für eine Dauer von über einem Jahr ohne Wiederholung der Sequenz der Code unaufhörlich laufen kann, so dass der hervorgebrachte Code geeignet ist eine Datenübertragung simultan zu identifizieren, codieren und decodieren, so dass eine Lifeübertragung von digitalen Signalen in codierter Form über die selbe Zeitspanne möglich ist. Die Programmierung kann auf eine solche Weise erfolgen, dass zwei oder mehrere der Codegenerierungseinheiten auf den selben Ausgangscode programmiert werden können, ohne, dass der Programmierer den Code kennt [Schaltung: Fig. 2, Spannungszustände an den einzelnen Pins der im Prototyp verwendeten Integrierten Schaltungen: Fig. 3].

Mag. René - Michael Cordes

FIGURENÜBERSICHT

Mag. René - Michael Cordes

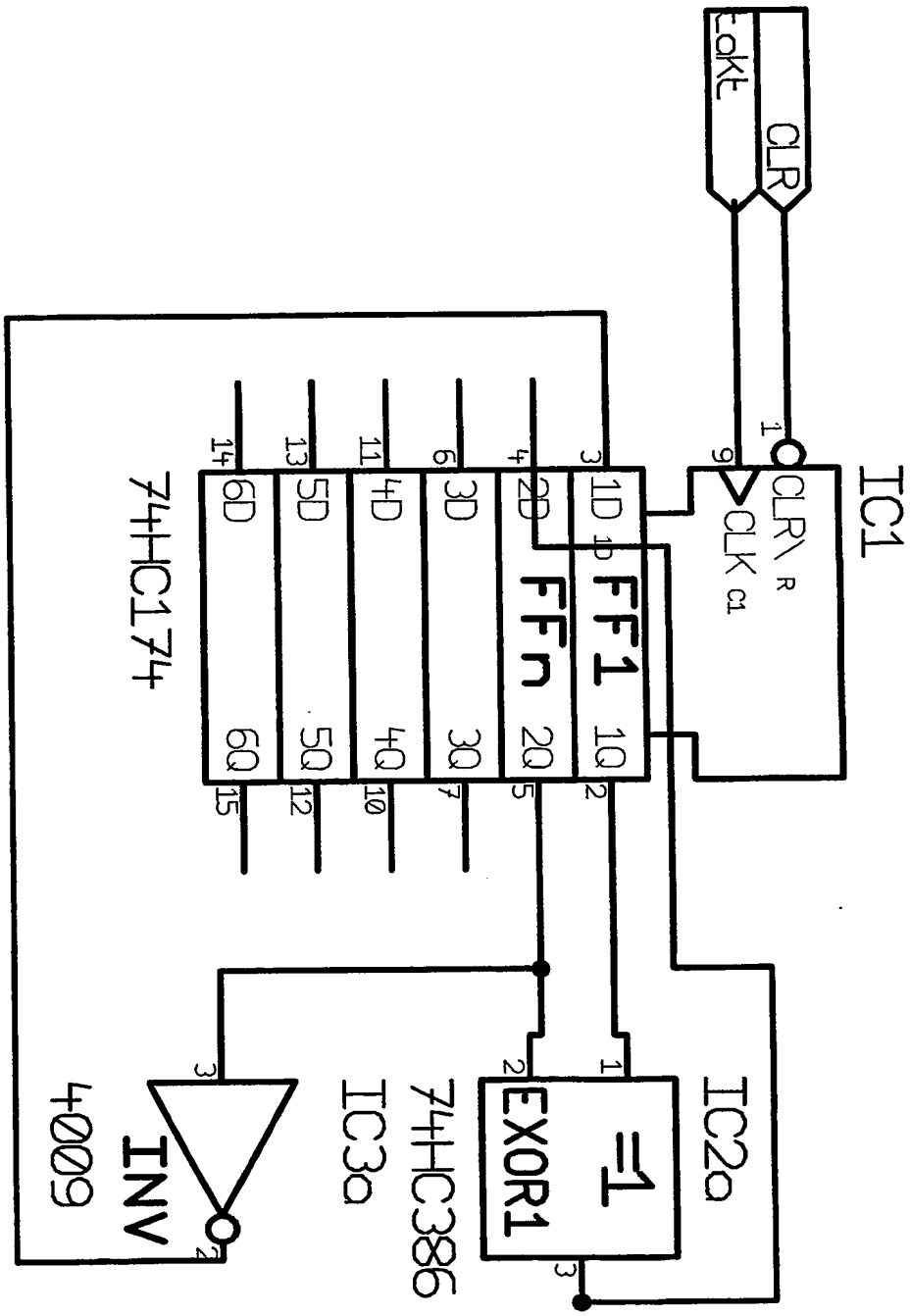


Fig. 1

Mag. René - Michael Cordes

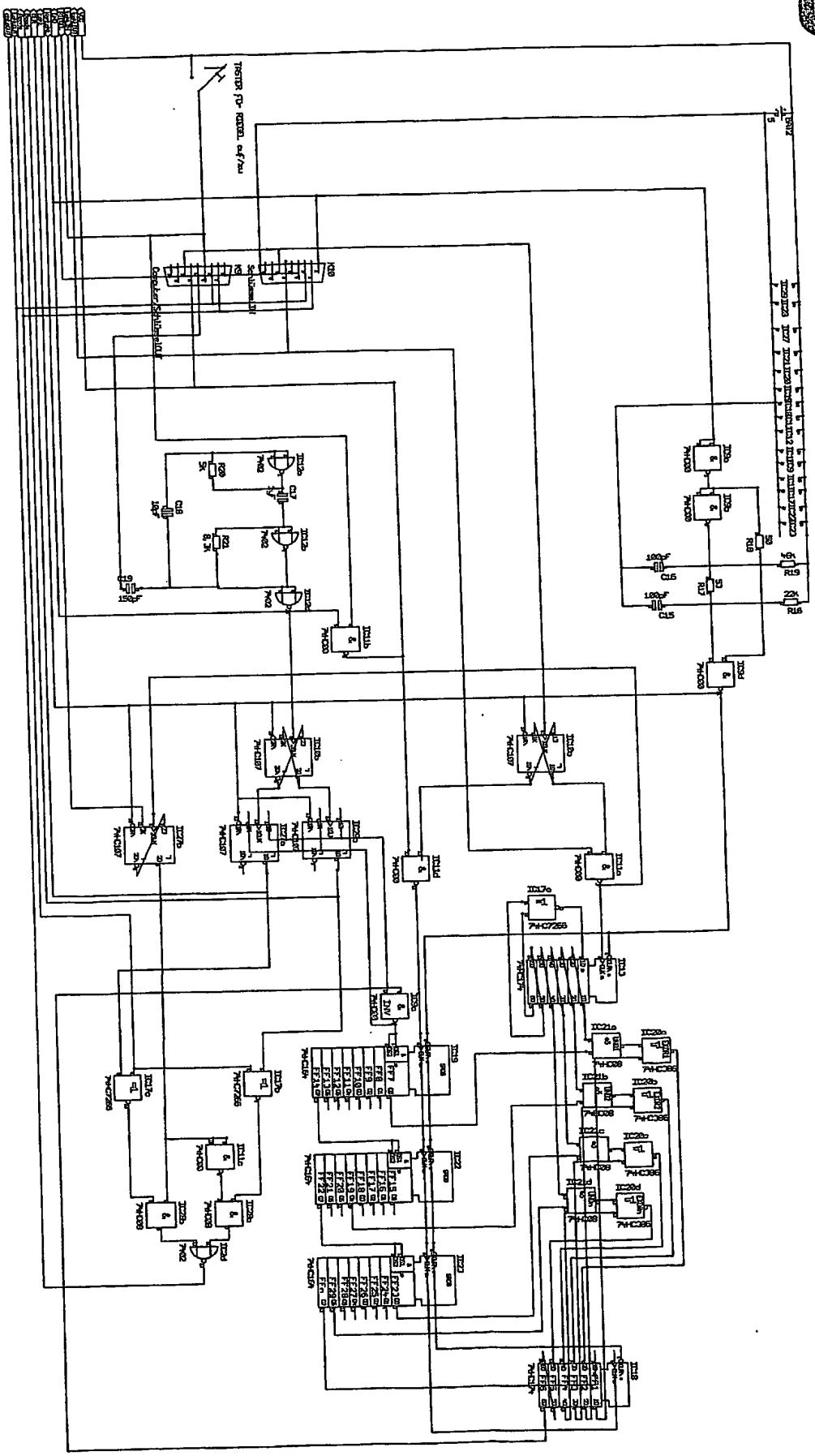


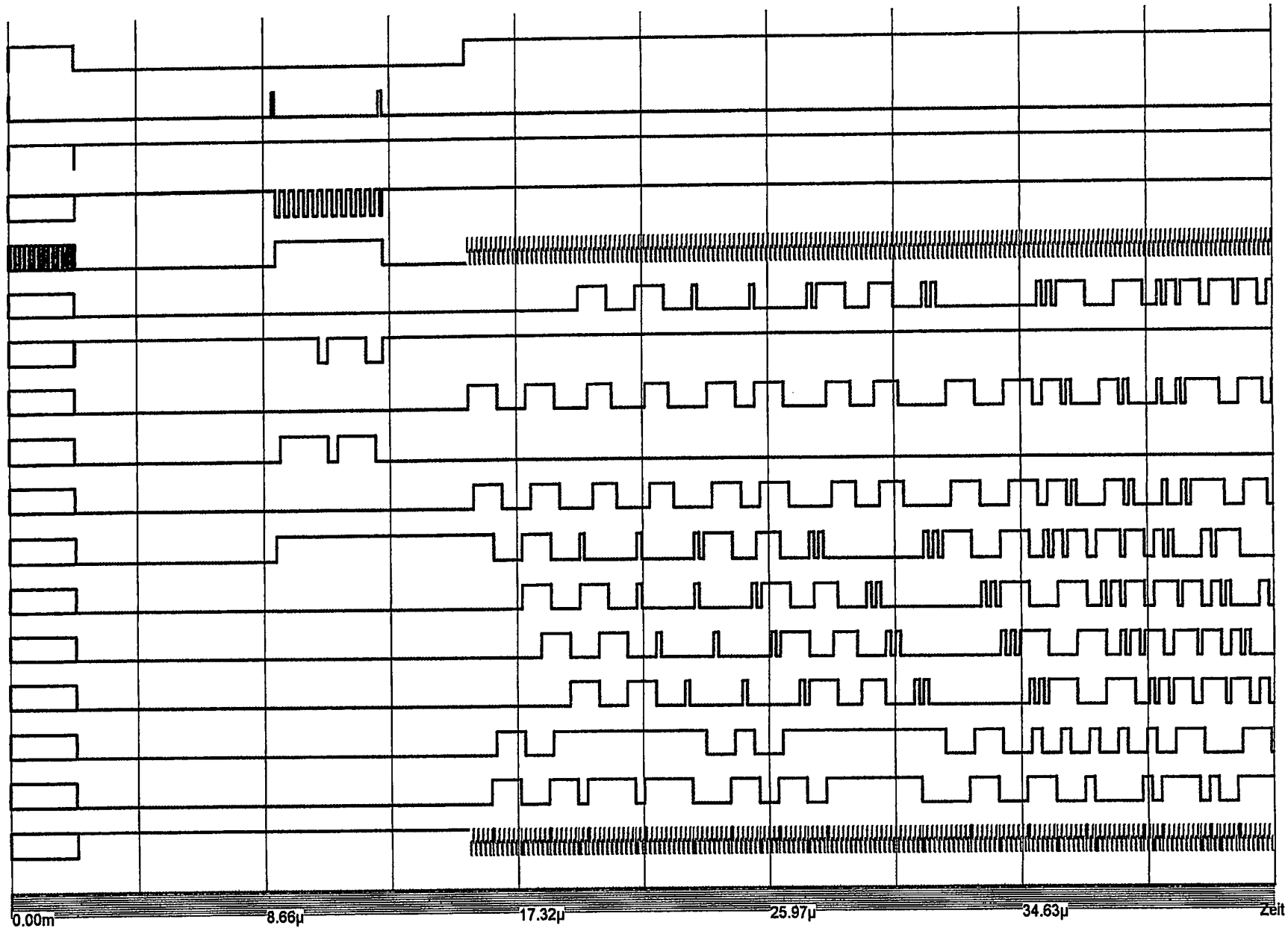
Fig. 2

A 338/2002

Fig. 3

Projekt: C:\Daten002\WIR\LC\Hardeware\LC5402 - LC111 - Fig 2.T3001
Datum: 03.03.2002
Zeit: 16:56:06

Mag. René - Michael Cordes



LC0111